COMPUTER SCIENCE THESIS PROPOSAL

19 Sept 2001

MEMORANDUM

From: Lieutenant Commander Ajax E. Helix
To: Curricular Officer, Computer Science Department
Via: (1) Thesis Advisor: Professor Persephone Wilson
(2) Academic Associate: Professor Wu
(3) Chairman, Computer Science Department: LCDR Chris Eagle

Subj: Thesis Report Number 1

Encl: (1) Milestone Plane for research and thesis completion
(2) Qualification Summary for Phibulous Shingle

1. Tentative Title of Proposed Thesis: "The Subversive Threat of Software Analysis"

2. General Area of Proposed Thesis Research: This research will demonstrate the potential risk posed to United States Information Systems from the analysis of subversion of a common operating system. The subversion risk will be explained and contrasted with the risks of penetration. In addition to the thesis report, a working example of a subverted operating system will be constructed.

3. Enclosure (1) is a milestone plan of dates and events for research and thesis completion.

4. I expect that my thesis will be unclassified.

5. I anticipate travel to discuss work with advisors and attend workshops and conferences pertinent to the thesis research.

6. Proposed Second Reader: Dr. Roger Schell

7. Second Reader's Signature: _____

_____
LCDR Ajax E. Helix

1. Approved and forwarded _____ _____
Thesis Advisor
Professor Persephone Wilson
Date

2. Approved and forwarded _____ _____
Academic Associate
Thomas Wu
Date

3. Approved and forwarded _____ _____
Chairman, Computer Science Department
LCDR Chris Eagle
Date

4. Approved and forwarded _____ _____
Curricular Officer
CDR Chris Lapacik
Date

A. General Information

1. Name: Lieutenant Commander Ajax E. Helix, USN
2. SMC: 1132
3. Curriculum: Computer Science, 368
4. Thesis Advisor: Professor Persephone Wilson
5. Second Reader: Dr. Phibulous Shingle
6. Academic Associate: Thomas Wu
7. Date of Graduation: March 2002

B. Area of Research

Develop a working demonstration of the security risk posed to U. S. information systems from subversion of an operating system. Linux will be selected for this demonstration. The kernel will be analyzed to determine logical places to add code that will enable an attacker to gain total control of the system. This modification will go beyond the typical "back door" in that control will be gained without the use of either the direct interfaces to the OS (e.g. logon) or through exploitation or penetration of the services it provides. In addition, the level of control will not be limited by the context of a particular service or process (e.g. process privilege levels).

C. Research Questions

1. What is the skill level required to implement this type of attack?
2. How much control can be provided and what are the limitations?
3. How difficult would it be to detect the presence of such a modification or to detect an attacker's use of the modification?

D. Discussion

Most efforts aimed at defending information systems are focused on preventing inadvertent disclosure or corruption of information or on thwarting attempts by a hacker to penetrate systems through exploitation of weaknesses present in systems and applications. The information systems of today are immensely complex and development progresses more rapidly than ever before. Obscure functionality, originally intended for troubleshooting and testing during development, can be accidentally left in the final product. As a result, vulnerabilities abound in these systems as evidenced by the level of activity in security mailing lists such as Bugtraq and the number of entries in MITRE's Common Vulnerabilities and Exposures (CVE) Repository. These vulnerabilities are the result of any combination of poor coding practices, lack of understanding on the part of developers in the areas of cryptography, network protocols, and poor design decisions in all phases of system development and maintenance. With such a high incidence rate of these types of defects, the question must be asked whether all of them are inadvertent. Could a shrewd developer or senior designer use this situation to intentionally insert a subtle artifice into a product in such a way that, if it ever were detected, attributing it to calculated malicious intent would be unlikely? The plausibility of the intentional insertion of artifices is demonstrated by the Easter Egg Archive (www.eeggs.com) which reports the presence of 2387 "easter eggs" present in computers and software as of the date of this memo. Easter eggs are typically 'bonus' functionally present in an application. These artifices are categorized in table 1.

Table 1. Computer Easter Eggs

Category	Total Eggs
Applications	743
Games	1336
Hardware	110
Operating Systems	133

Other	65

The vast majority of these involve developer lists examples of which are as elaborate as a flight simulator or auto racing game, which display the names of developers as the user moves through the simulation. These artifices are commonly triggered by obscure sequences of operations in the application.

It is obvious that hidden artifices are not only possible but, in fact, do exist. They are an extremely viable option for subverting a system (in particular, an operating system) in a way that could give an attacker total and virtually undetectable control over it.

E. Scope of Thesis

Demonstration of a hypothetical adversaries capability to create a working operating system containing an artifice that provides total control over system resources and data.

F. Methodology

This thesis will involve analysis of a commercial of-the-shelf operating system for the most appropriate method for inserting an artifice. Criteria to be considered are scope of control, risk of detection during use, risk of discovery during code review, and risk of neutralization of the artifice through upgrades and patches. A thorough knowledge of operating systems, C programming, x86 assembly language, and reverse engineering will be required.

G. Chapter Outline

I. Introduction
   - A. Historical Background
   - B. Contrast with other attack methods.
   - C. Purpose of Study
   - D. Organization of Paper

II. Justification as a Valid Attack Technique
   - A. Survey of recent vulnerabilities and their sources
   - B. Easter Eggs

III. Operating System Selection and High Level Design Considerations
   - A. Operating System Selection and Impact of This Decision
   - B. Selecting the Method of Subversion
   - C. Artifice Design and Integration Into the OS
   - D. Interface to the Artifice - Exploitation

IV. Discussion of Detectability of Artifices

V. Limiting the Risk of Artifices in Software Systems.

VI. Conclusions

H. Schedule

1. Study of Linux Kernel Design	OCT 2001
2. Determination of Artifice Location	NOV 2001
3. Design of Artifice and Interface	NOV 2001
4. Coding and Testing	DEC 2001
5. Draft Thesis	JAN 2002
6. Final Thesis Submission/Signature	FEB 2002

I. Benefits of Study

The threat from subversion has been known for quite some time. However, little attention has been paid to this threat most likely due to the fact that penetration incidents are far more noticeable. This study will demonstrate the extent to which an artifice can be used against the United States and highlight the difficulty in detecting its presence and defending against it.

J. Anticipated Travel/Funding Requirements

Anticipate moderate local travel. Moderate level of funding for books, hardware, and software (approx $3000).

K. Preliminary Bibliography.

Myers, Philip A., Subversion: The Neglected Aspect of Computer Security. Naval Postgraduate School Thesis, June 1980.

Beck, M. Ed. Linux Kernel Internals. 2nd Ed., Addison Wesey, New York, 1998.

Bovet, D. P., and Cesati, M., Understanding the LINUX Kernel: From I/O Ports to Process Management. O'Reilly, 2000

Aivazian, T., Linux Kernel 2.4 Internals, Ver. 1.2. The Linux Documentation Project. Available On-line: http://www.linuxdocs.org, 2001

Rusling, D. A., The Linux Kernel, ver. 0.8-3. The Linux Documentation Project. Available On-line: http://www.linuxdocs.org, 2001